1

Developing Shared Ontologies in Multi-agent Systems

Dean Jones

Department of Computer Science, University of Liverpool, U.K.

dean@csc.liv.ac.uk

ABSTRACT Effective communication of knowledge between agents needs to be based on a common understanding of the terms contained in messages. A shared ontology forms the basis for specifying the meaning of terms used by communicating agents. The construction of a shared ontology must therefore form one of the main stages in the development of a multi-agent system.

In this paper, we suggest that the design of a shared ontology is intrinsically linked to the development of the relevant communicating agents and cannot be separated out as an isolated stage in itself. In this article, we describe the skeleton of a methodology for the development of a particular class of agent network. This description locates the development of the shared ontology in the overall process.

1.1 Introduction

The KRAFT project is concerned with the transformation and integration of knowledge located in distributed heterogeneous resources (Gray *et al.*, 1997). This integration requires the development of a network of multiple communicating agents. Currently, there is little in the way of principled methodologies that can be used in the development of multi-agent systems. One of the reasons for this could be the variety of software artefacts that are referred to as agents. This multiplicity suggests that no single general methodology will apply in the development of every multi-agent system. However, if the development of multi-agent systems is to progress into an engineering practice, methodological approaches to the development of such systems need to be available. In this paper, we describe the approach taken in the design of KRAFT networks, paying particular attention to the role and development of domain ontologies, with the aim of generalising this approach over multi-agent systems designed for the integration of heterogeneous resources.

In the next section, we address in more detail the role of shared ontologies in multi-agent systems, particular in relation to a KRAFT network. In section 3, we describe the methodology that is applied in the development of a KRAFT network. A summary and a discussion of further relevant issues are given in section 4.

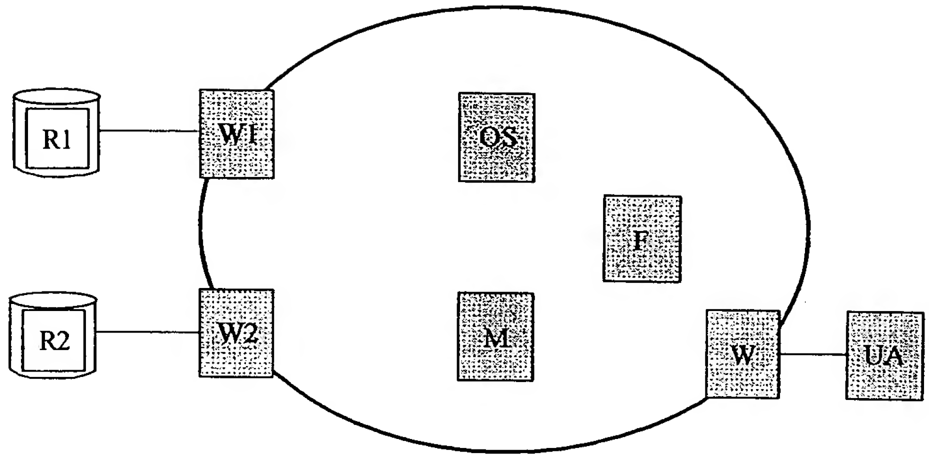

FIGURE 1.1. A simple KRAFT Network

1.2 KRAFT Networks and Shared Ontologies

The starting point for the development of a KRAFT network is (a) set of existing resources, which are typically knowledge-based systems and/or databases, and (b) some problem(s) to be addressed using the information located in the resources. A network of communicating agents is then developed that can utilise the information located in the resources in order to solve the problem(s). As stated in Gruber (1995), for agents to engage in knowledge-level communication, three conventions are required: (a) common representation format, (b) agent communication protocol, and (c) specification of the content of shared knowledge. In a KRAFT network, these requirements are satisfied by (a) Constraint Interchange Format (CIF), a language for the expression of constraint-based knowledge, (b) Constraint Communication and Query Language (CCQL), a variant of KQML (Finin et al., 1997), and (c) one or more shared ontologies. Such a shared ontology forms the basis for a common understanding of the content of messages that are passed between the agents in the network.

Five classes of agent have been identified as being required in the development of a KRAFT network. As agents are distinguished on the basis of their function, each will be introduced by describing their (typical) role in the resolution of a query. This description is represented in Fig. 1. Firstly, the query is entered via a *user agent* (UA), which is essentially an interface to the network that may or may not be associated with a resource. The query is then passed to a *wrapper* (W), which provides translation services between resources and the network (see Visser *et al.*, 1997; 1998 for more details). Wrappers ensure that all messages passed to the network is a well-formed KRAFT message, i.e. they conform to the three conventions outlined above, and will perform the necessary translations. The wrapper sends the query in the form of a KRAFT message to a *facilitator* (F), which provides various routing services, such as content-based routing. When a resource or agent joins a KRAFT network, it must register its capabilities with a facilitator. On the basis of these advertisements, the facilitator attempts to identify a resource or agent that can answer the query. To

perform this identification, the facilitator may communicate with an *ontology server* (OS) in order to obtain information about the shared ontology for intelligent routing purposes. Simple queries may require only that the query be forwarded to one or more resources (R1 and R2), where wrappers (W1 and W2) again provide translation services. More complex queries will be forwarded to a *mediator* (M), which will attempt to decompose a query into simple queries that can be answered by one or more resources. If the mediator cannot fully decompose the query into simple queries, it will ask the facilitator to identify other mediators that can provide the relevant query decomposition services. Once the query has been fully decomposed and all sub-queries have been resolved, the initial mediator combines the results and returns them to the user agent. Note that we do not necessarily restrict ourselves to these five agent types. Any other kind of agent, collectively referred to as *auxiliaries*, can potentially participate in a KRAFT network.

As may be evident from the above description, ontologies have a fundamental role in the KRAFT architecture. A shared ontology is used as the basis for defining the mapping functions that enable the wrappers to perform semantic translation services. A shared ontology is also used to answer questions about concepts in the domain in order that facilitators can provide intelligent content-based routing services. In this article, the method adopted in the development of a shared ontology and the relationship of this process to the overall design of the network is described.

A shared ontology specifies the meaning of terms that form the content of queries within the KRAFT "ringfence" (see Fig. 1). Mapping functions are applied by wrappers to effect translations between the vocabularies of the individual resources and the vocabulary defined by the shared ontology. The terms used in an ontology are either (a) *primitive*, i.e. not defined, or (b) *non-primitive*, i.e. defined in terms of other (primitive and/or non-primitive) terms. As the meaning of primitive terms cannot be identified from an ontology they can be given different and inconsistent interpretations. As the definition of non-primitive terms ultimately relies on the interpretation of the primitive terms, these too can be interpreted inconsistently. One method of reducing the possibility of such inconsistent interpretations is to use select the primitives used from a standard vocabulary. Several formally-defined standard vocabularies, commonly referred to as *top-level ontologies*, have been defined which can be used for this purpose, examples being SENSUS (Knight *et al.*, 1995), WordNet (Miller *et al.*, 1990) and the CYC upper layer (Lenat and Guha, 1994). Top-level ontologies are fairly comprehensive as they define a large number of the terms in some language. This allows all of the concepts defined in a shared ontology to be linked to the top-level ontology as all primitive terms can be selected from a top-level ontology, which provides a consensus on the meanings of primitive terms.

Note that, although we are currently investigating the use of multiple shared ontologies in KRAFT networks, for the purposes of this paper we concentrate on the simplest case where there is only one shared ontology.

1.3 The Methodological Development of KRAFT Networks

The approach adopted in the development of a KRAFT network is termed reductionist by Jennings and Campos (1996), as the problem being addressed is decomposed into sub-problems, which are then assigned to one or more individual agents. Although

Jennings and Campos (1996) suggest that the reductionist approach fails to exploit the full potential of the agent paradigm, implicit in our approach is what Jennings and Campos (1996) term the benevolent system assumption, which states that all agents will willingly perform all tasks requested of them and volunteer their services if they have sufficient free capacity. In KRAFT we are mainly concerned with investigating the integration of constraint-based knowledge. We have adopted a multi-agent paradigm as the best means of addressing this problem but we are not specifically interested in investigating the possible interactions that can occur among a network of agents.

The following description is not a comprehensive methodology for the development of a KRAFT network. Here we are attempting to specify the stages involved in the development of a shared ontology and the relationships between these steps and the development of the KRAFT network as a whole. We believe that the development of a shared ontology and the development of the agent network are mutually dependent processes. Therefore we describe in some detail the development of a shared ontology and pass over some stages with only brief outlines.

1.3.1 REQUIREMENTS ANALYSIS

As in more conventional software engineering, the first stage in the development of a multi-agent system is the specification of the problem(s) being addressed. The requirements analysis should identify and specify (a) the users of the network, (b) the queries that the network needs to be able to respond to, (c) the resources that are available to the network in resolving the queries, (d) the expected output of the system.

1.3.2 TASK DESCRIPTION

Once problem has been fully specified, the tasks that are required to solve it can be described. This is done in the following stages:

(a) *task decomposition*: the decomposition of tasks will be based on (a) the resources that are available for the solution of the problem, and (b) any recognised methods of solving the problem(s). As in the DESIRE approach (Brazier *et al.*, 1997), a task is either *complex*, i.e. it can be described by one or more subtasks, or *primitive*, i.e. it cannot be decomposed further. Task decomposition is initiated by identifying a *problem-solving method* (PSM) that will enable it to be solved. A PSM defines how a task will be solved in terms of its subtasks and the information required. A PSM is identified for each complex subtask and the process continues until the main task has been fully decomposed into primitive tasks. Note that not all PSMs used in this approach would strictly be identified as such in more traditional knowledge engineering. For example, issuing primitive queries in a pre-defined sequence (see Fig. 2) would not commonly be labelled a PSM.

(b) *task allocation*: the tasks can now be allocated to individual mediators according to the following principles. Starting with the top-level task:

(i) assign a complex task to a mediator.

task: *assess* possibility of student *s* moving to year *y* at University *u*.
PSM: *generative and test*: iteratively generate prerequisites, *p*, for a valid set of modules *m* for year *y* at *u* and test against history of *s*

- **task:** *generate* prerequisite selection *p* for modules *m* for year *y* at university *u*
  PSM: *coordinated primitives*: issue primitive queries in pre-defined manner
    - **task:** *generate n*, number of modules for *y* at *u*
      PSM: *primitive* DB query
    - **task:** *generate m*, set of *n* modules for year *y* at *u*
      PSM: *primitive* DB query
    - **task:** *generate* prerequisite selection *p* for set of modules *m*
      PSM: *primitive* DB query
- **task:** *generate* history, *h* for student *s*
  PSM: *primitive* DB query
- **task:** *asssess* whether history *h* of student *s* satisfies prerequisites *p*
  PSM: *primitive operation*: compare *h* and *p*

FIGURE 1.2. Task analysis for the KRAFT student transfer system

(ii) assign primitive subtasks to the mediator identified in (i).

(iii) assign complex subtasks to a new mediator.

(iv) repeat process for each complex subtask.

Each complex task needs to be allocated to a single mediator in order that the flow of information during the execution of a PSM can be controlled. It is possible that all of the identified tasks could be allocated to a single mediator. However, each complex task is assigned to a different mediator in order to take advantage of the benefits of distributed problem solving. As an example, consider the task decomposition given in Fig. 2. This is an analysis of the student transfer problem that was addressed by the KRAFT prototype system, which will determine whether a student on a degree course at a university can transfer to the named year of a similar course at a different university. Various heterogeneous resources were connected to the network that store the academic records of students currently taking degree courses and information on degree courses and modules at three different institutions (Aberdeen, Cardiff and Liverpool). The user enters the name of a student at a university, the name of another university and a year to transfer into. The system would then determine whether the named student satisfies the prerequisites for the given year at the relevant institution. Fig. 2 represents the decomposition of this problem into its primitive components. From the analysis, it is suggested that two mediators be developed to address the problem.

(c) *task resolution*: the communications that will take place in resolving the specified problems should be described. A 'walk-through' document will describe, step by step, the messages that will be sent to, from and across the network in resolving the required tasks. From this, the terms that are essential to the communication of the necessary queries and their corresponding results can be identified. To facilitate this stage of the system design, we have defined a generic walkthrough that can be instantiated for each problem addressed.

1.3.3 SHARED ONTOLOGY DEVELOPMENT

Following the task description, the definition of the shared ontology can begin. This process consists of the following stages:

(a) *ontology scoping*: most ontology development methodologies define a scoping phase as one of the first stages in the process (Grüninger and Fox, 1995; Uschold, 1996). For the shared ontology in a network of communicating agents, the minimal scope is the set of terms that are necessary to support the required communications. The set of terms identified in the task description as necessary for the communication of the required queries and results provides the initial scope of the shared ontology. These terms are initially described in the form of a data dictionary that consists of a list of terms and natural language definitions for them. It should also be remembered that the terms used to request mediator functions (termed *capabilities*) will form the content of some inter-agent messages.

(b) *domain analysis*: given that change is an integral feature of large-scale computer systems (Wiederhold, 1995), it is reasonable to assume that the tasks that will need to be supported by a shared ontology will change. One solution to this is to define a minimal shared ontology and allow if to be refined and extended when new functionality is added to the network. However, the fact that a shared ontology is situated in a network can cause problems for such maintenance, even for monotonic extensions. For example, a monotonic extension may introduce redundancy into a shared ontology, which allows alternative mapping functions to be defined between ontologies. In order that the network can easily incorporate additional functionality, the shared ontology should not be limited to the minimal scope defined during stage (iii). Rather, we pursue a domain-led strategy (Paton *et al.*, 1991) whereby the shared ontology fully characterises the field of knowledge that the problem being addressed is situated in. For example, in the student transfer problem, this methodology requires that the domain of higher education be defined by the shared ontology. This approach facilitates the maintenance and extensibility of the network as it allows additional functionality, in the form of new mediators, to be incorporated without having to extend the ontology. However, it does not ensure that refinement of the shared ontology will not be required.

(c) *ontology formalisation*: once the domain has been fully characterised, the shared ontology needs to be formally represented. Many subjective decisions will be made in the formalisation of an ontology (see e.g., Jones and Paton, 1998; Jones *et al.*, 1998a for details of such decisions). However, this process can be based on a principled approach. For example, the ontological engineer should be aware of the ontological commitments that a particular representational formalism makes. The definition of terms as classes, relations, functions, etc., requires careful consideration of these ontological distinctions. It is helpful to consider principles such as those of Guarino *et al.* (1994), which help to distinguish predicates that identify an individual as a kind from those that assert some property as being true about a kind. Such 'meta-level' distinctions are necessary analytical tools in the formalisation of a body of knowledge.

(d) *introduction of top-level ontology*: as stated earlier, relationships should be defined between the primitive terms in a shared ontology and a top-level ontology. Some of the primitive terms will be defined in the top-level ontology in the correct sense and

other primitive terms will not. For these undefined terms, the relationship to the top-level terms will be some other relation, such as a subtype relationship.

When using the SENSUS ontology in the development of a domain-specific ontology (Swartout *et al.*, 1997), a set of "seed" terms are selected as relevant domain-specific concepts and these are linked to the SENSUS ontology. For a KRAFT shared ontology, the only terms that will be linked to the top-level ontology are the primitive terms. The difference between the two approaches is best explained by stating that Swartout *et al.* (1997) use the top-level ontology as part of the domain analysis stage, whereas in KRAFT the domain analysis is performed prior to introduction of the top-level ontology. It should also be remembered that defining the relationships between the shared ontology and a top-level ontology is a design process. The ontological engineer will need to make various decisions, such as selecting the correct sense of a term defined in the top-level ontology.

Note that following the development of the shared ontology, the terms used in the task description may require some small refinement.

1.3.4 DOCUMENTATION

It is common in software engineering to have a single stage during which documentation is produced. In this approach, documentation describing the agents and the ontologies is produced at all stages of the development. However, it is useful to identify a single stage during which this material is collated and organised.

1.3.5 MAINTENANCE

There are various issues in the maintenance of multi-agent systems that need to be addressed. Due to lack of space we address only one issue concerning the shared ontology. When a new mediator agent is introduced into the network, a new requirements specification should be produced which describes the new problems to be addressed. The content of the communications in this document should be specified, whenever possible, in terms of the existing shared ontology. This will minimise the extension of the shared ontology that the introduction of a new mediator agent might otherwise require. This is facilitated by the domain-led strategy adopted in the initial design of the shared ontology.

1.4 Conclusions

This paper has outlined the principles that we have adopted in the development of KRAFT multi-agent systems. We have described how the analysis of the tasks involved in solving a particular problem is necessary before the shared domain ontology can be developed. We have also described the stages involved in the design and development of a shared ontology.

The decomposition of the main problem that is being addressed enables complex tasks to be associated with individual PSMs. This will benefit from the development of a library of problem-solving methods, which in turn will further facilitate software re-

use. Also, the allocation of tasks to mediator agents is a fundamental stage in the design process. The distribution of complex tasks to individual mediators further facilitates re-use as it simplifies the selection of previously built mediators in the development of a new network.

Re-use would also be improved though use of a library of ontologies. However, we are not optimistic that this is currently a practical possibility as there are many issues that need to be resolved. The principles on which a library is organised need to be investigated in order that the selection of an ontology can be made in a sensible way. There is the related problem of how ontologies should be described in a library such that the ontology that most closely suits a particular need can be identified. Additionally, in a recent survey of ontological engineering approaches, (Jones *et al.*, 1998a), we noted that ontology development methodologies are commonly task-oriented, as the starting point in development is the intended use of the ontology. This severely limits the potential for ontology re-use. It seems plausible that, for shared ontologies, reuse is a necessary sacrifice in favour of other factors such as maximising the potential for communication.

One of main conclusions of this work results from the fact that the functionality that a shared ontology is required to support is not static. This requires that the shared ontology is designed such that this change can be effectively managed. By basing the design of the shared ontology on a domain analysis rather than on the tasks performed by a network, the likelihood of being required to update the shared ontology is minimised.

1.5 Acknowledgements

Thanks are due to the other members of the KRAFT project at Liverpool who have contributed to this article. We would also like to express our gratitude to our collaborators at the Universities of Aberdeen and Cardiff and at BT.

1.6 Bibliography

Brazier, F.M.T., Dunin-Keplicz, B.M., Jennings, N.R. and Treur, J. (1997) "DESIRE: Modelling Multi-agent Systems in a Compositional Formal Framework", *International Journal of Cooperative Information Systems*, **6**(1), 67-94.

Finin, T., Labrou, Y. and Mayfield, J. (1997) "KQML as an Agent Communication Language", in J. Bradshaw (ed.) *Software Agents*, MIT Press, Cambridge, MA.

Gray, P.M.D., Preece, A.D., Fiddian, N.J., Bench-Capon, T.J.M., Shave, M.J.R., Azarmi, N., Wiegand, M.E., Ashwell, M.A., Beer, M.D, Cui, Z., Diaz, B.M., Embury, S.M., K.Y. Hui, A.C. Jones, D.M. Jones, G.J.L. Kemp, E.W. Lawson, Lunn, K., Marti, P., Shao, J. and Visser, P.R.S, (1997) "KRAFT: Knowledge Fusion from Distributed Databases and Knowledge Bases", in *Proc. DEXA-97*, September 1-5th, Toulouse, France.

Gruber. T.R. (1995) "Towards Principles for the Design of Ontologies used for Knowledge Sharing", *International Journal of Human-Computer Studies*, **43**(5/6), 907-928.

Grüninger, M. and Fox, M.S. (1995) "Methodology for the Design and Evaluation of Ontologies", *IJCAI-95 Workshop on Basic Ontological Issues in Knowledge Sharing*, August 19-20th, Montreal.

Guarino, N., Carrara, M and Giaretta, P. (1994) "An Ontology of Meta-Level Categories", in J. Doyle, E. Sandewall and P. Torasso (eds.) *Principles of Knowledge Representation and Reasoning: Proceedings of the 4th International Conference*, Morgan Kaufmann, San Mateo, CA.

Jennings, N.R. and Campos J.R. (1996) "Towards a Social Level Characterisation of Socially Responsible Agents", in *IEE Proc. on Software Engineering*, **144**(1), 11-25.

Jones, D.M. and Paton, R.C. (1998) "Some Problems in the Formal Representation of Hierarchical Knowledge", to be presented at *International Conference on Formal Ontology in Information Systems - FOIS'98*, Trento, Italy, June 6-8th.

Jones, D.M., Visser, P.R.S. and Paton, R.C. (1998a) "Addressing Biological Complexity to Enable Knowledge Sharing" to be presented at *AAAI Workshop on Knowledge Sharing Across Biological and Medical Knowledge Based Systems*, Madison, Wisconsin, July 26-27th.

Jones, D.M., Bench-Capon, T.J.M. and Visser, P.R.S. (1998b) "Methodologies for Ontology Development", in *Proc. IT&KNOWS Conference, XV IFIP World Computer Congress*, Budapest, August.

Knight, K., Chander, I., Haines, M., Hatzivassiloglou, V., Hovy, E.H., Iida, M., Luk, S. K., Whitney, R. and Yamada K. (1995) "Integrating Knowledge Bases and Statistics in MT", *Proc. of the 14th International Joint Conference on Artificial Intelligence*, Montreal, Quebec.

Lenat, D.B. and Guha, R.V. (1994) "Enabling Agents to Work Together", *Communications of the ACM*, **37**(7), 127-142.

Miller, G.A., Beckwith, R., Fellbaum, C., Gross D. and Miller, K.J. (1990) "Introduction to WordNet: An On-line Lexical Database", *International Journal of Lexicography*, **3**(4), 235-244.

Paton, R.C., Nwana, H.S., Shave, M.J.R., Bench-Capon, T.J.M. and Hughes, S. (1991) "Foundations of a Structured Approach to Characterising Domain Knowledge", *Cognitive Systems*, **3**(2), 139-161.

Swartout, W.R., Patil, R., Knight, K. and Russ, T. (1997) "Towards Distributed Use of Large-Scale Ontologies", *AAAI-97 Spring Symposium on Ontological Engineering*, Stanford University, March 24-26th

Uschold, M. (1996) "Building Ontologies: Towards A Unified Methodology", *Proc. Expert Systems '96*, Cambridge, December 16-18th.

Visser, P.R.S., Jones, D.M., Bench-Capon, T.J.M. and Shave, M.J.R. (1997) "An Analysis of Ontology Mismatches: Interoperability versus Heterogeneity", *AAAI-97 Spring Symposium on Ontological Engineering*, Stanford University, March 24-26th.

Visser, P.R.S., Jones, D.M., Bench-Capon, T.J.M. and Shave, M.J.R. (1998) "Assessing Heterogeneity by Classifying Ontology Mismatches", to be presented at *International Conference on Formal Ontology in Information Systems - FOIS' 98*Trento, Italy, June 6-8th.

Wiederhold, G. (1995) "Modeling and System Maintenance" in M.P. Papazoglou (ed.) *Proceedings OOER'95:Object-Oriented and Entity Relationship Modelling*; Springer Lecture Notes in Computer Science, Vol. 1021, 1-20.